Claims 29-31

Claims 29-31 depend from claim 26 and are allowable for the same reasons. In addition, claim 29 patentably distinguishes Sadre in that it recites further limitations relating to the recited task control mechanism none of which, as discussed above, are neither disclosed or suggested by Sadre.

Claims 31 and 32

Claim 31 depends from claim 26 and is allowable for at least the same reasons. In addition, it recites a step of converting structured textual language into a processor-independent pseudo code. The passage from Sadre relied on in rejecting this claim nowhere refers or even suggests conversion to processor-independent pseudo code. For this additional reason, the claim is submitted to be allowable. Claim 32 is allowable for similar reasons.

Claim 34

Claim 34 depends from claim 26 and is allowable for the same reasons. The claim also recites that additional graphical elements are generated by converting user-defined structured text subprograms of the textual language into graphical elements comprising function interfaces of the corresponding structured text subprograms. The language of Sadre relied on in rejecting this claim may mention the words "structured text program," but it decidedly does not disclose generating graphical elements from those programs. For at least this additional reason, claim 33 should be allowed.

Claim 37

Claim 37 depends from claim 34 and is allowable for the same reasons. Still further, it recites that a user can switch between structured textual language, contact plan and function plan as forms of representation for formulation conditions. Sadre, at column 19, lines 14-20 relied on in rejecting this claim, may refer to switching between "sequential program" and "associated continuous program." This, however, does not disclose or suggest the recited claim language; for at least this further reason claim 37 should be allowed.

Claim 48

Claim 48 depends from claim 31 and is allowable for the same reasons. It recites, furthermore, that the re-translation back into motion control flowchart representation is possible by means of marks in the textual language. The language of Sadre relied on to support the rejection fails to disclose this limitation. That language, referring to Figure 10 (which is also deficient) refers to a structured text translation of a sequential function chart. No re-translation back into a flowchart representation is disclosed or suggested. Claim 48 is therefore allowable for this further reason.

Claim 49

Claim 49 depends from claim 26, and is allowable for the same reasons, but recites that steps a-c are triggered in a collective step. Since those steps are not shown by Sadre, nor are they shown in a collective step. Figures 23A and 23B do not appear to show any collective steps, but rather a series of steps – which do not happen to be the steps recited in the claim. Claim 49 is submitted to be patentable over the art of record for this additional reason.

Claim 50

Claim 50, rejected on grounds analogous to those raised against claim 32, is submitted to be patentable for the same reasons.

Claim 52

Claim 52 is an independent claim rejected on grounds analogous to those set forth in the rejection of claim 26.

Applicant respectfully submits that neither the text nor figures of Sadre does not disclose or suggest the limitations of preparing a plurality of debugging processes for programming code for the industrial controller, where the code has a plurality of code levels, conducting debugging for the plurality of processes and displaying the processes on respective interfaces. For at least this reason, claim 52 is submitted to be patentable over the art of record.

Claim 53

Claim 53 depends from claim 52 and is allowable for the same reasons. In addition, claim 53, wherein at least a subset of the plurality of debugging processes corresponds to respective ones of the plurality of code levels and the steps of displaying debugging processes comprises displaying at least a subset of the debugging processes on respective ones of the interfaces. This limitation, said to be shown by Sadre at col. 17, line 62 – col. 18, line 15, does not actually disclose a plurality of code levels, or the display of corresponding debugging processes on respective displays. For this additional reason, claim 53 is submitted to be unpatentable.

Claim 54

Claim 54 depends from claim 52 and is allowable for the same reasons. It also recites that the plurality of code levels comprises a pseudo-code level and a debugging process is prepared for that level. The passage relied upon in rejecting this claim points to language apparently relating to breakpoints in structured text. It does not, however, refer to a plurality of code levels, pseudo code as one of those levels, or the preparing of a debugging process at the pseudo code level. For this further reason, claim 54 should be allowed.

Claim 55

Claim 55, which is rejected on grounds similar to those raised against claim 26, is submitted to be allowable for the reasons set forth above in connection with claim 52.

Claim 56

Claim 56, rejected on grounds similar to those raised in rejecting claim 32, is submitted to be patentable on the basis of the arguments raised above for the patentability of claims 32 and 52.

IV. Rejection of Claim 39 Under 35 U.S.C. § 103(a)

Claim 39 is rejected over Sadre, further in view of U.S. Patent No. 4,837,722 to Sara (Sara). This rejection, however, is misplaced. First, claim 39 depends from claim 38 and is submitted to be patentable on the same grounds.

A rejection under 35 U.S.C. § 103(a) requires the establishment of a *prima facie* case that the claimed subject matter, including all claim elements, would have been obvious to a person having ordinary skill in the art on the basis of either a single prior art reference or more than one reference properly combined. As no such *prima facie* case has been established for these claims, Applicants respectfully traverse these rejections, as set forth more fully below.

The combination of thc applied references, moreover, is improper. Sara, relating to digital color television reproduction, is manifestly not analogous to Sadre. Even assuming, without conceding, that Sara supplies the deficiencies of Sadre, one of ordinary sill in the field of industrial software could not reasonably be expected to combine these references without the benefit of hindsight reliance on the present application.

Although the office action admits that Sadre does not disclose a parallel branch in which individual commands are initiated in a given interpolator cycle within a respective parallel branch. Moreover, Sara neither discloses nor suggests a programming language command of a motion control flowchart nor initiating a parallel branch within such a flowchart. For these reasons, no prima facie case of obviousness has been established. Accordingly, claim 39 is submitted to be allowable over the art of record.

V. Rejection of Claim 40 Under 35 U.S.C. § 103(a)

Claim 40 has been rejected as unpatentable over Sadre, further in view of U.S. Patent No. 6,295,606 to Messerges ("Messerges").

A rejection under 35 U.S.C. § 103(a) requires the establishment of a *prima facie* case that the claimed subject matter, including all claim elements, would have been obvious to a person having ordinary skill in the art on the basis of either a single prior art reference or more than one reference properly combined. As no such *prima facie* case has been established for these claims, Applicants respectfully traverse these rejections, as set forth more fully below.

As described in Applicants past two amendments, claim 40 depends from allowable claim 26 and is therefore patentable. Moreover, the combination of references lacks propriety. Messerges relates to cryptography and, more particularly, to leakage attacks on microelectronic assemblies. One of ordinary skill in the field of industrial software could not have been expected to look to this field to identify solutions in his or her own field, nor has any motivation been identify as to why, much less how, one would combine Sadre and Messerges, or how one would pick and choose among their disclosures without relying, impermissibly, on Applicants' disclosure. Even assuming, without conceding, the references could be combined, their combination would neither disclose nor suggest all of the limitations of the claimed invention.

The passage of Messerges relied upon to support the rejection fails to describe or suggest function blocks, much less setting parameters for the function blocks by mask input in the display associated with the motion control flowchart. For these various reasons, claim 40 is submitted to be patentable over the art of record.

CONCLUSION

Upon entry of this Amendment, claims 26, 29-56 are pending in the application. Applicants submit that the claims, for the reasons set forth above, are in condition for allowance. Reconsideration and allowance are therefore respectfully requested. The Commissioner is authorized to charge any necessary fees to Deposit Account No. 23-1703.

Dated: January 5, 2006

Respectfully submitted,

[illegible]

Scott T. Weingaertner
Reg. No. 37,756
Attorney for Applicants

Customer No. 007470
White & Case LLP
Direct Line: (212) 819-8404

Amendments to the Claims

The following listing of claims will replace all prior versions and listing of claims in the application.

1-25. (Canceled)

26. (Currently amended) A method for debugging a program for an industrial controller, the industrial controller having an engineering system including an editor and further having a run time system, wherein graphical elements are linked using the editor to form a motion control flowchart that can be visualized on a display, the graphical elements corresponding to respective tasks, the method comprising the steps of:

a) preparing a debugging process for the industrial control program based on the flowchart;

b) assigning a suspend command to each graphical element of the flowchart;

c) commencing the debugging process for the industrial control program;

d) continuing the debugging process until a suspend command is reached;

e) displaying the location of the flowchart element corresponding to the suspend command;

f) continuing a task corresponding to a graphical element of the flowchart, that has been suspended by a suspend command, using a task control mechanism of the run-time system; and

~~f)~~ g) proceeding to the next possible suspend command.

27. (Canceled)

28. (Canceled)

29. (Previously presented) The method according to claim 26, wherein the task control mechanism of the run time system comprises breakpoint debugging and variables that can be

pre-assigned by the user in the engineering system, further comprising the step of pre-assigning variables corresponding to breakpoints.

30. (Previously presented) The method according to claim 29 wherein the variable pre-assignments in the task control mechanism are performed by programs of the run time system other than the task control mechanism.

31. (Previously presented) The method according claim 26, comprising the steps of:
   a) generating a structured textual language from the flowchart;
   b) converting the structured textual language into a processor-independent pseudo-code;
   c) loading thc processor-independent pseudo-code into a controller;
   d) converting the processor-independent pseudo-code into executable processor code.

32. (Previously presented) The method according to claim 26, wherein a debugging interface is available to a user at levels comprising at least two of the group consisting of the structured textual language level, the pseudo-code level, and the processor code level.

33. (Previously presented) A method according to claim 26, wherein programming language commands are provided in the flowchart editor as a function of configuration of hardware associated with an industrial controller.

34. (Previously presented) The method according to claim 26, wherein additional graphical elements are generated in the motion control flowchart representation by converting user-defined structured text subprograms of the textual language into graphical elements comprising function interfaces of the corresponding structured text subprograms.

35. (Previously presented) The method according to claim 34, wherein the generated graphical elements are used as language elements of the motion control flowchart.

36. (Previously presented) The method according to claim 26, wherein structured text according to IEC 6-1131 is used as a structured textual language.

37. (Previously presented) The method according to claim 36, wherein a user can switch between structured textual language, contact plan and function plan as forms of representation for formulation conditions.

38. (Previously presented) The method according to claim 26, wherein at least one of the group consisting of a loop and a parallel branch is present as a programming language command in the motion control flowchart view.

39. (Previously presented) The method according to claim 38, wherein a parallel branch is initiated and wherein individual commands are initiated in a given interpolator cycle within a respective parallel branch.

40. (Previously presented) The method according claim 26, further comprising function blocks, wherein parameters can be set for the function blocks by mask input in a display associated with the motion control flowchart.

41. (Previously presented) The method according to claim 26, further comprising function blocks, wherein the function blocks are combined into modules that in turn are presented as function blocks in a display associated with the motion control flowchart.

42. (Previously presented) The method according to claim 41, wherein modules are interleaved in the display associated with the motion control flowchart.

43. (Previously presented) The method according to claim 41, wherein the function blocks for the allocation of variables in the display associated with the motion control flowchart comprise multiple instructions.

44. (Previously presented) The method according to claim 41, wherein the function blocks representing functions that require a given period of time comprise advance conditions in the display associated with the flowchart.

45. (Previously presented) The method according to claim 26, wherein the graphical elements of the flowchart are positioned automatically.

46. (Previously presented) The method according to claim 26, wherein the graphical elements of the flowchart are linked together automatically.

47. (Previously presented) The method according to claim 26, wherein the flowchart is displayed in a form comprising one of the group consisting of a reduced form and an enlarged form.

48. (Previously presented) The method according to claim 31, wherein re-translation back into motion control flowchart representation is possible by means of marks in the textual language.

49. (Previously presented) The method according to claim 26, wherein steps a) through c) are triggered in a collective step.

50. (Previously presented) The method according to claim 26, wherein during processing of the flowchart program a currently processed graphical element is displayed.

51. (Previously presented) The method according to claim 26, wherein the different code levels comprise at least one of the group consisting of structured text code, processor independent pseudo-code, and object code.

52. (Previously presented) A method for debugging a program for an industrial controller, the industrial controller having a plurality of code levels associated with at least one of an engineering system and a run time system associated with the industrial controller, wherein graphical elements are linked using an editor to form a motion control flowchart that can be visualized on a display, the method comprising the steps of:

a) preparing a plurality of debugging processes for programming code for the industrial controller having the plurality of code levels, the programming code associated with the flowchart;

b) conducting debugging for the plurality of debugging processes; and

c)displaying the debugging processes on respective ones of a plurality of debugging interfaces.

53. (Previously presented) The method according to claim 52, wherein the programming code comprises a plurality of code levels, at least a subset of the plurality of debugging processes corresponds to respective ones of the plurality of code levels, and the step of displaying debugging processes comprises displaying at least a subset of the debugging processes on respective ones of the plurality of debugging interfaces.

54. (Previously presented) The method according to claim 52, wherein the plurality of code levels comprises a pseudo-code level and a debugging process is prepared for the pseudo-code level.

55. (Previously presented) A method for debugging a program for an industrial controller, the industrial controller having a plurality of code levels associated with at least one of an engineering system and a run time system associated with the industrial controller, wherein graphical elements are linked using an editor to form a motion control flowchart that can be visualized on a display, the method comprising the steps of:

a) preparing a debugging process for programming code for the industrial controller having the plurality of code levels, the programming code associated with the flowchart;

b) conducting debugging for the debugging process; and

c) displaying the debugging process at a plurality of levels.

56. (Previously presented) The method according to claim 55, wherein the plurality of levels comprise at least one selected from the group consisting of the structured textual language level, the pseudo-code level, and the processor code level.